Maximum Weight Matching Dispatching Scheme in Buffered Clos-network Packet Switches

Roberto Rojas-Cessa, *Member, IEEE,* Eiji Oki, *Member, IEEE,* and H. Jonathan Chao, *Fellow, IEEE*

***Abstract*—The scalability of Clos-network switches make them an alternative to single-stages switches for implementing large-size packet switches. This paper introduces a cell dispatching scheme, called maximum weight matching dispatching (MWMD) scheme, for buffered Clos-network switches. The MWMD scheme is based on a maximum weight matching algorithms for input-buffered switches. This paper shows that, with request queues in the buffered Clos-network architecture, the MWMD scheme is able to achieve a 100% throughput for independent admissible traffic, without allocating any buffers in the second stage and without expanding the internal bandwidth. As a practical scheme, a maximal oldest-cell-first matching dispatching (MOMD) scheme is also introduced. MOMD shows that using a finite number of iterations in the dispatching scheme, the throughout under unbalanced traffic pattern can be high.**

***Index Terms*—Packet switch, clos-network, dispatching, throughput, maximum-weight matching**

I. Introduction

It is well known that single-stage switches have limited scalability, in terms of the number of ports. This limited scalability is the result of the limited number of connection pins that a switch chip can allocate. The three-stage Clos-network [1], [2] switch can provide a higher degree of scalability than single-stage switches, because of the smaller number of switch chips needed for a large size switch.

We can categorize the Clos-network switch architecture into two types: bufferless and buffered. A bufferless Clos-network switch has no memory in any stage. To avoid contention in any stage, scheduling needs to be performed at the input ports prior to sending the packets through the switch. This approach has the advantage of simplifying the design of the switch modules. However, the matching process may be complex and require a long resolution time. Scheduling of bufferless Clos-network switches can also be used in optical switches where switch reconfiguration may not have to change as fast as per time slot. There are several studies on scheduling for bufferless Clos-network switches [7], [8], [9]. They, however, are out of the scope of this paper.

Here, we assume that variable-length packets are segmented into several fixed-sized packets, or cells, when they arrive. Cells are switched through the switch fabric, and reassembled into packets before they depart. The time to transmit a cell through the switch is called time slot.

Roberto Rojas-Cessa is with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, University Heights, Newark NJ, 07102. Email: rojasces@njit.edu

Eiji Oki is with NTT Network Innovation Laboratories, NTT Corporation, 3-9-11 Midori-cho Musashino-shi, Tokyo 180-8585 Japan. E-mail: oki.eiji@lab.ntt.co.jp

H. Jonathan Chao is with the Department of Electrical and Computer Engineering, Polytechnic University, Brooklyn, NY, 11201. Email: chao@poly.edu

One way to ease the complexity of scheduling in Clos-network switches is by allocating memory in the first and third stages. In this way, if contention for an internal link occurs, loser cells are stored in the buffers in the first stage modules. These switches can be referred to as buffered Clos-network switches. As the memory technology evolves, the memory amount that can be embedded into a chip is no longer a strict limitation. Within buffered Clos-network switches, we can consider two groups: with and without buffers in the second-stage modules.

A gigabit ATM (asynchronous transfer mode) switch using buffers in the second-stage was presented in [3]. In this architecture, every cell is randomly distributed from the first-stage to the second-stage modules to balance the traffic load in the second-stage. Implementing buffers in the second-stage modules resolves contention among cells from different first-stage modules [15]. However, it requires a re-sequencing function at the third-stage modules, because the buffers in the second-stage modules cause an out-of-sequence problem.

A three-stage switch with buffers in the first and third stages and bufferless second stage is called a buffered Clos-network switch. In [4], an ATM switch was developed. This approach does not suffer from the out-of-sequence problem. Since there are no buffers in the second-stage modules to resolve contention, dispatching cells from the first stage to the second stage becomes an important issue. A random dispatching (RD) scheme is used for cell dispatching from the first stage to the second stage [4], as adopted in the case of the buffered second-stage modules in [3]. However, RD is not able to achieve a high throughput unless the internal bandwidth is expanded, because the contention at the second stage cannot be avoided. To achieve 100% throughput for uniform traffic by using RD, the internal expansion ratio is set to about 1.6 when the switch size is large [2], [4]. This expansion makes a high-speed switch difficult to implement in a cost-effective manner.

It has been shown that it is possible to provide 100% throughput under uniform traffic without expanding the internal bandwidth on a buffered Clos-network switch with a round-robin-based dispatching scheme [2]. Moreover, diverse dispatching schemes have been proposed to reduce the average cell delay [5], [11] under uniform traffic. However, real traffic patterns are not only uniform, but a wide variety of admissible traffic patterns, including those with nonuniform distributions.

One question arises: Is it possible to achieve a 100% throughput under independent admissible traffic, without allocating any buffers in the second stage to avoid the out-of-sequence problem and without expanding the internal bandwidth?

This paper proposes a cell dispatching scheme, called

maximum weight matching dispatching (MWMD) scheme, for buffered Clos-network switches. The MWMD scheme is based on the maximum weight matching algorithm for input-buffered switches [6]. It is known that for independent admissible traffic, a maximum throughput of 100% in an input-buffered switch is achievable by using a maximum weight matching algorithm [6]. We show that by the considering request queues in a buffered Clos-network switch and the MWMD scheme it is possible to achieve a 100% throughput for independent admissible traffic, without allocating any buffers in the switch modules at the second stage and without expanding the internal bandwidth. Furthermore, we introduce an iterative dispatching scheme, maximal oldest cell first matching dispatching (MOMD), based on the oldest-cell-first (OCF) scheme [13] for single-stage input-buffered switches, and show that it can provide 100% throughput under our nonuniform traffic model, called unbalanced.

This paper is organized as follows. Section II describes our switch model. Section III discusses the MWMD scheme. Section IV describes the MOMD scheme. Section V shows the performance study. Section VI presents the conclusions.

II. Buffered Clos-network Switch Model

Figure 1 shows a buffered Clos-network switch. The first stage consists of k input modules (IMs), each of which has an $n \times m$ dimension. The second stage consists of m buffer-less central modules (CMs) each of which has a $k \times k$ dimension. The third stage consists of k output modules (OMs), each of which has an $m \times n$ dimension.

The terminology used in this paper is as follows:
$IM(i)$: $(i+1)$th input module, where $0 \le i \le k-1$.
$CM(r)$: $(r+1)$th central module, where $0 \le r \le m-1$.
$OM(j)$: $(j+1)$th output module, where $0 \le j \le k-1$.
n: number of input/output ports in each IM/OM, respectively.
k: number of IMs/OMs.
m: number of CMs.
$IP(i,h)$: $(h+1)$th input port (IP) at $IM(i)$, where $0 \le h \le n-1$.
$OP(j,l)$: $(l+1)$th output port (OP) at $OM(j)$, where $0 \le l \le n-1$.
$VOMQ(i,j)$: virtual output-module queue at $IM(i)$ that stores cells destined for $OM(j)$.
$RQ(i,j,r)$: Request queue (RQ) at $IM(i)$ that stores requests of cells destined for $OM(j)$ through $CM(r)$. $RQ(i,j,r)$ also keeps the waiting time $W(i,j,r)$, which is the number of slots a head-of-line (HOL) request has been waiting.
$L_I(i,r)$: output link of $IM(i)$ that is connected to $CM(r)$.
$L_C(r,j)$: output link at $CM(r)$ that is connected to $OM(j)$.
$\lambda(i,h,j,l)$: arrival rate at $IP(i,h)$ for $OP(j,l)$.
$\lambda(i,j)$: arrival rate at $IM(i)$ for $OM(j) = \sum_h \sum_l \lambda(i,h,j,l)$.

An $IM(i)$ has k virtual output-module queues (VOMQs) to eliminate Head-Of-Line (HOL) blocking. A VOMQ is similar to a virtual output queue (VOQ), which is used in input-buffered switches [13], [6]. When a cell enters $VOMQ(i,j)$, the cell request is randomly distributed and stored in $RQ(i,j,r)$ among m request queues. A request in $RQ(i,j,r)$ is related to $VOMQ(i,j)$, but is not related to a specific cell in $VOMQ(i,j)$. Every time a request in $RQ(i,j,r)$ is granted for transmission, one cell is dequeued from $VOMQ(i,j)$ in a FIFO manner. A VOMQ can receive at most n cells from n input ports and can send at most m cells to m CMs in one time slot.[1] Each $IM(i)$ has m output links. Each output link $L_I(i,r)$ is connected to each $CM(r)$.

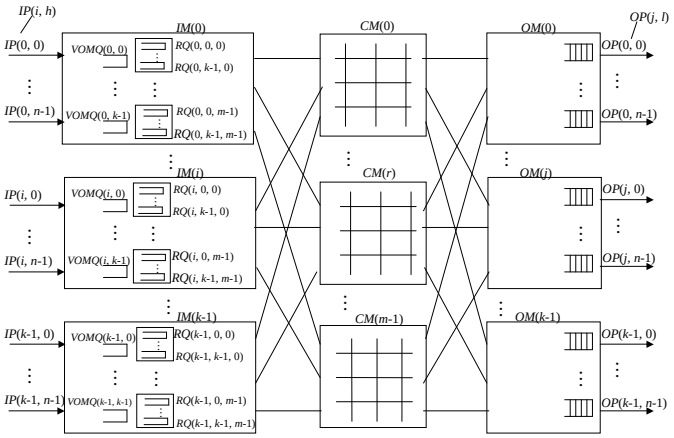

Fig. 1. Clos-network switch with virtual output-module queues (VOMQs) in the input modules

A $CM(r)$ has k output links, $L_C(r,j)$, connected to $OM(j)$. An $OM(j)$ has n output ports, each of which is denoted as $OP(j,l)$ and has an output buffer. Each output buffer receives at most m cells in one cell time slot, and each output port at the OM forwards one cell in a first-in-first-out (FIFO) manner to the output line.

Scalability, in terms of the port speeds, has also been of research interest. A single-stage parallel packet switch resembles the switch fabric of a Clos-network switch (i.e., the set of IMs, CMs, and OMs) because of the way the internal links are connected in the latter. Parallel switches consider using switch planes running at lower speed than the external connection links to provide high-speed ports [14]. We consider that Clos-network switches use internal links transferring data at the same speed as the external links. Moreover, following the principle of the parallel switch, Clos-networks switches can also use parallel switching planes to provide high port speeds. However, the discussion about increasing the port speed in a Clos-network switch is out of the scope of this paper.

III. Maximum Weight Matching Dispatching (MWMD) Scheme

The MWMD scheduler consists of m subschedulers, each of which is denoted as $S(r)$, as shown in Figure 2. Subscheduler $S(r)$ selects up to k requests from k^2 RQs, where corresponding cells to the selected RQs are transmitted through $CM(r)$ at the next time slot. In $S(r)$, k RQs: $RQ(i,0,r)$, ..., $RQ(i,k-1,r)$, are the requests from $IM(i)$ to all OMs through $CM(r)$, and k RQs: $RQ(0,j,r)$, ..., $RQ(k-1,j,r)$,

[1] An L-bit cell must be written to and read from a VOMQ memory in a time less than $\frac{L}{Cn}$ and $\frac{L}{Cm}$, respectively, where C is a line speed. For example, when $L = 64 \times 8$ bits, C=10 Gbit/s, and $n = m = 8$, $\frac{L}{Cn} = \frac{L}{Cm}$ is 6.4 ns. This is feasible when we consider 2-port memories with current available CMOS technologies.

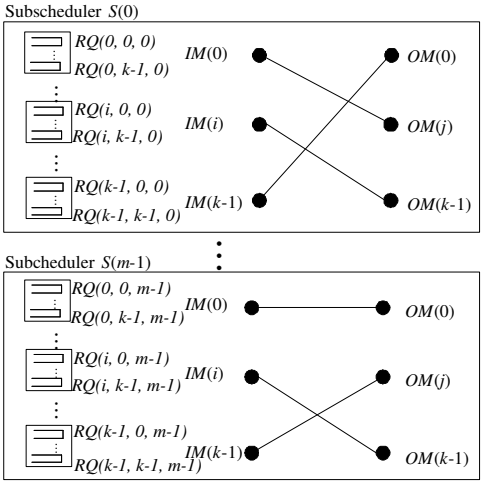

Fig. 2. Maximum Weight Matching Dispatching (MWMD) Scheduler

are the requests from all IMs to $OM(j)$ through $CM(r)$. $S(r)$ selects one request from each $IM(i)$ and one request to each $OM(j)$. $S(r)$ finds an appropriate match according to its scheduling algorithm as an input-buffered scheduler does.

In the request selection in $S(r)$ at each time slot, a maximum weight matching algorithm, the oldest-cell-first (OCF) algorithm [6], is employed. $RQ(i,j,r)$ keeps the waiting time $W(i,j,r)$, which is the number of slots a HOL request has been waiting. $S(r)$ finds a match $M(r)$ at each time slot, so that $\sum_{(i,j)\in M(r)} W(i,j,r)$ is maximized. Note that each $S(r)$ behaves independently and concurrently. $S(r)$ uses only k^2 $W(i,j,r)$ to find $M(r)$. $S(r)$ and $S(r')$, where $r < r'$, do not exchange any information to find $M(r)$ and $M(r')$, respectively.

When $RQ(i,j,r)$ is granted by $S(r)$, the HOL request in $RQ(i,j,r)$ and a cell in $VOMQ(i,j)$ are dequeued. The dequeued cell is one of the HOL cells in $VOMQ(i,j)$. Here, the number of dequeued HOL cells is equal to the number of granted requests by all $S(r)$s.

In order to send cells in sequence in an output buffer at $OP(j,l)$, we adopt a simple rule as follows. Consider that Y RQs for the same (i,j) pair, which are $RQ(i,j,r_1)$, $RQ(i,j,r_2)$, ..., $RQ(i,j,r_Y)$, where $r_1 < r_2 < r_Y$, are granted by more than one subscheduler. Y cells from the HOL in $VOMQ(i,j)$ are transmitted to $OM(j)$ through $CM(r_1)$, $CM(r_2)$, ..., $CM(r_Y)$ at the same time slot. The yth cell from the HOL in $VOMQ(i,j)$ goes through $CM(r_y)$. Note that each request in $RQ(i,j,r)$ is related to $VOMQ(i,j)$, but is not related to a specific cell in $VOMQ(i,j)$, as described in Section II. When more than one cell goes to an output buffer at $OP(j,l)$, a cell from $CM(r)$ enters the output buffer earlier than the cell from $CM(r')$, where $r < r'$. As a result, cells are sent in sequence.

A. 100% throughput by MWMD

We prove that MWMD achieves 100% throughput for all admissible independent arrival processes without internal bandwidth expansion, i.e., with $n = m$.

Theorem 1: MWMD achieves 100% throughput for all admissible independent arrival processes without expansion of the internal bandwidth.

Proof:

With this architecture, we use the rate matrix of the arrival process as:

$$\Lambda \equiv [i,h,j,l],$$

where the associate rate vector is

$$\underline{\lambda} \equiv (\lambda_{0,0,0,0}, \ldots, \lambda_{0,0,0,n-1}, \ldots, \lambda_{0,0,k-1,0}, \ldots, \lambda_{0,0,k-1,0}, \ldots, \lambda_{0,n-1,0,0}, \ldots, \lambda_{k-1,0,0,0}, \ldots, \lambda_{k-1,n-1,k-1,n-1}).$$

We assume that our admissible input traffic conditions,

$$\sum_i \sum_h \lambda(i,h,j,l) < 1 \quad \text{and} \quad \sum_j \sum_l \lambda(i,h,j,l) < 1, \quad (1)$$

are satisfied.

The arrival process $A_i(t)$, which is the aggregated cell arrivals of n input ports at $IM(i)$ that are destined to $VOMQ(i,j)$, is stationary and ergodic. The arrival matrix representing the sequence of arrivals $A(t) \equiv [A_{i,j}(t)]$, where, at time t:

$$A_{i,j}(t) = \begin{cases} g, 1 \le g \le n & \text{if arrival(s) occurs at } VOMQ_{i,j} \\ 0 & \text{otherwise,} \end{cases} \quad (2)$$

and the associated rate vector is

$$\underline{A}(t) \equiv (A_{0,0}(t), \ldots, A_{0,k-1}(t), \ldots, A_{k-1,k-1}(t))^T.$$

The service matrix, $\Gamma(t) \equiv [\Gamma_{i,j}(t)]$ indicates which VOMQs are serviced at time t:

$$\Gamma_{i,j}(t) = \begin{cases} g, 1 \le g \le n & \text{if } VOMQ_{i,j} \text{ gets service at } t \\ 0 & \text{otherwise,} \end{cases} \quad (3)$$

and the associate vector $\underline{\Gamma}(t)$ is similarly defined as $\underline{A}(t)$.

However, $\Gamma(t)$ may not be used as a permutation matrix. To overcome this, consider using request queues, $RQ(i,j,r)$, as described in Section III. As the MWMD scheme, performed by the rth subscheduler considers the RQs related the rth CM for dispatching cells through $CM(r)$, the set of RQs can be considered independent of the others. They can be separated as a local set of RQs, denoted as $LRQ(i,j)$, for each $CM(r)$. Each LRQ has a corresponding arrival request matrix $\Delta(t) \equiv [\Delta_{i,j}(t)]$. The request arrival matrix is such that

$$\Delta_{i,j}(t) = \begin{cases} 1 & \text{if an arrival to } LRQ_{i,j} \text{ occurs at time } t \\ 0 & \text{otherwise,} \end{cases} \quad (4)$$

as RQs accept in average one request from input i to output j each time slot, and the RQ distribution is independently and informingly distributed. Its associated vector, $\underline{\Delta}(t)$, is similarly defined.

The service received by LRQs is determined by the service matrix $\Theta(t) \equiv \Theta_{i,j}(t)$, where:

$$\Theta_{i,j}(t) = \begin{cases} 1 & \text{if } LRQ_{i,j} \text{ is served at time } t \\ 0 & \text{otherwise,} \end{cases} \quad (5)$$

and the associated vector, $\underline{\Theta}(t)$, is similarly defined. Therefore, the LQRs and CMs are equivalent to the VOQs of an input-buffered switch, considering $\Delta(t)$ and $\Theta(t)$ as the arrival

and service processes. The service received by LRQs is the result of MWMD, where only the HOL requests at LRQs are matched.

From (1), we obtain that

$$\sum_i \lambda(i,j) = \sum_i \sum_h \sum_l \lambda(i,h,j,l) < n, \qquad \text{and}$$
$$\sum_j \lambda(i,j) = \sum_j \sum_h \sum_l \lambda(i,h,j,l) < n. \tag{6}$$

Here, $\sum_i 1 = \sum_j 1 = n$ is used. By dividing (6) by m,

$$\frac{1}{m}\sum_i \lambda(i,j) < \frac{n}{m} \quad \text{and} \quad \frac{1}{m}\sum_j \lambda(i,j) < \frac{n}{m}. \tag{7}$$

By using $n = m$, we obtain

$$\frac{1}{m}\sum_i \lambda(i,j) < 1 \quad \text{and} \quad \frac{1}{m}\sum_j \lambda(i,j) < 1. \tag{8}$$

Therefore, the total service received by $VOMQ(i,j)$ is $\sum_r \Theta_{i,j}(t)$. If the received service by $\Theta_{i,j}(t)$ for each CM and its LRQ set is such that the RQs are stable, then MWMD is stable.

Note that $\frac{1}{m}\lambda(i,j)$ is the request arrival rate for $RQ(i,j,r)$. In addition, since the cell arrival process at $VOMQ(i,j)$ is independent, a request arrival process to $RQ(i,j,r)$ is also independent. This is because, when a cell enters $VOMQ(i,j)$, its request is randomly distributed and stored among m request queues. As MWMD is based on the maximum-weight matching scheme, Lemma 7 in [6] can be applied.

Therefore, the subscheduler model in MWMD is equivalent to the scheduler model in the maximum wieght matching (MWM) of the input-buffered switch model defined in [6]. Using Lemmas 8-10 and Theorem 4 in [6], where the OCF-based MWM algorithm has the queue occupancy stable, achieving 100% throughput in an input-buffered switch for all admissible and independent arrival processes, a subscheduler in MWMD provides 100% throughput. As all subschedulers perform similarly, MWMD provides 100% throughput for all admissible and independent arrival process with $n = m$. ■

IV. Maximal Oldest cell first Matching Dispatching (MOMD)

In this section, we introduce the maximal oldest-cell first matching dispatching (MOMD) scheme with multiple memory access (i.e., queues can dispatch up to m cells) as a maximal dispatching scheme with lower complexity than MWMD. MOMD uses the switch model presented in Section II. There are k VOMQs at each IM, each denoted as $VOMQ(i,j)$, and m request queues, RQs, each associated with a VOMQ, and denoted as $RQ(i,j,r)$. MOMD has distributed arbiters in IM and CM. In $IM(i)$, there are m output-link arbiters. In $CM(r)$, there are k arbiters, each of which corresponds to $OM(j)$. The VOMQs and distributed link arbiters are shown in Figure 3. When a cell enters a $VOMQ(i,j)$, the cell gets a time stamp assigned, which is entered along with a request to $RQ(i,j,r)$, where r is randomly selected. We consider link arbiters for the output links at IMs and CMs. To determine the matching between $VOMQ(i,j)$ and the output link $L_I(i,r)$,

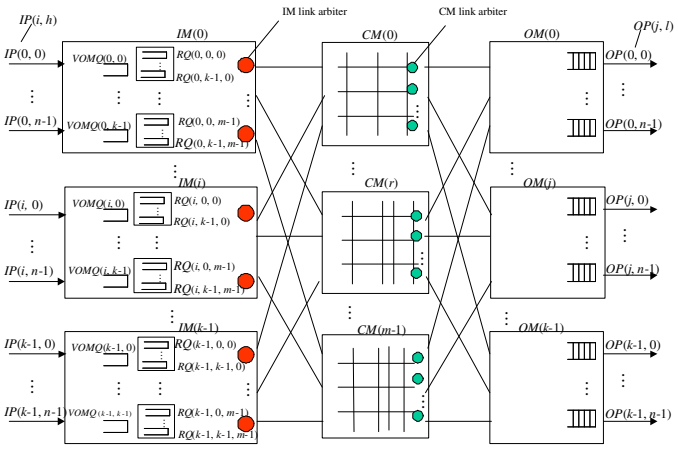

Fig. 3. Distributed arbiters in a buffered Clos-network with MOMD

the output links arbiters select a VOMQ (through their RQs) based on time stamps.

The scheme is described as follows:

Phase 1: Matching within IM

Step 1: A non-empty $RQ(i,j,r)$ sends a request to the unmatched output link arbiter associated to $L_I(i,r)$. The request includes a weight, which is the time stamp of the HOL request.

Step 2: Each output-link arbiter $L(i,j)$ chooses one RQ request by selecting the oldest time stamp. If a tie occurs, the output link arbiter selects the $RQ(i,j,r)$ by selecting the VOMQ with the largest index j. The output link arbiter sends the grant to the selected RQ and VOMQ.

Phase 2: Matching between IM and CM

Step 1: After phase 1 is completed, $L_I(i,r)$ sends the request to $CM(r)$ belonging to the selected VOMQ. At $CM(r)$, each arbiter associated with $OM(j)$ chooses one request by selecting the oldest time stamp. Ties are broken by selecting, among the older requests, the largest index i. The arbiter at $CM(r)$ sends the grant to $L_I(i,r)$ of $IM(i)$.

Step 2: If an IM receives a grant from a CM, the IM sends a (HOL) cell from that VOMQ at the next time slot. Note that up to m cells, at the head of line, may be dispatched from a VOMQ.

MOMD dispatches cells as in MWMD to get cells arrive the output buffer at $OP(j,l)$ in sequence.

The described scheme refers to a single iteration between IM and CM. To perform more iterations, the unmatched VOMQs, links and CM arbiters are considered and the rejected requests are inhibited in the remaining iterations between IM and CM in the time slot.

V. Performance Evaluation of MOMD

We show the delay and throughput performance of the dispatching schemes presented above in 64×64 switches, where $n = m = k = 8$, using the RD [4], CRRD [2], and MOMD schemes. Note that MOMD does not use the iteration in IM, denoted as iIM, as in the CRRD scheme. MOMD uses request queues that simplify the process, such that one link arbiter selects one of m request queues. However, we consider that a number of iterations between IM and CM in all schemes,

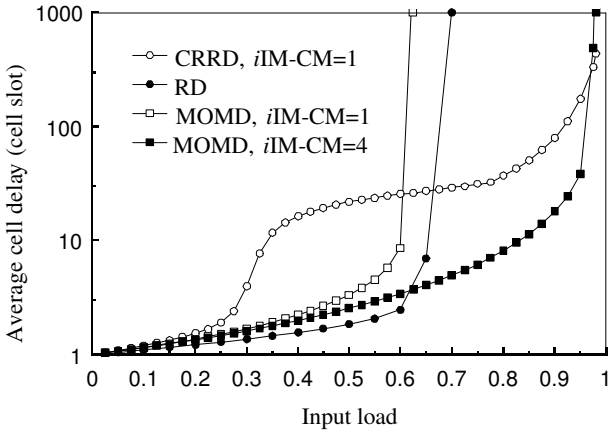

Fig. 4. MOMD under bernoulli uniform traffic ($n = m = k = 8$)) in the input modules

denoted as iIM-CM, is used.[2] We note that MOMD cannot achieve a 100% throughput under uniform traffic with a single IM-CM iteration. However, MOMD will get high throughput by increasing the number of IM-CM iterations, as shown in the figure. In the switch under simulation, where $n = m = k = 8$, the number of iteration to provide 100% throughput is four (iIM-$CM = 4$). The simulation shows that the round-robin based scheme, CRRD, is more effective under uniform traffic than MOMD, as CRRD achieves high throughput with one iIM-CM=1.

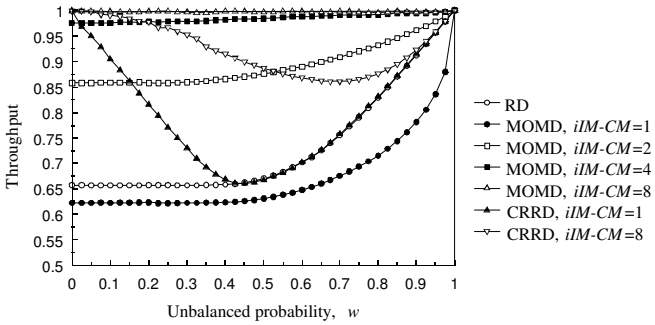

Fig. 5. MOMD under bernoulli unbalanced traffic ($n = m = k = 8$)

We also consider a nonuniform traffic pattern, called unbalanced [16]. The unbalanced traffic pattern has one fraction of traffic with uniform distribution and the other fraction w of traffic destined to the output with the same index number as the input. When $w = 0$, the traffic is uniform; when $w = 1$ the traffic is totally directional. We evaluate RD, CRRD, and MOMD under this traffic pattern. To obtain the best performance, CRRD adopts $iIM = m$ (i.e., the number of iterations performed in IM to match the VOQs used in CRRD and output links [5]) for any number iIM-CM. Figure 5 shows that RD, with one iteration, and CRRD, with multiple iIM-CM iterations, cannot offer 100% throughput under unbalanced traffic. Although CRRD with iIM-$CM = 8$ improves the throughput, it is not close to 100%. The reason for that is that CRRD, based on round-robin selection, tries to allocate the same service for all queues, independently of the HOL cell age or queue load in each time slot. MOMD, as it uses weight-based selection, provides higher throughput as the number of IM-CM iterations increases. MOMD with iIM-$CM = 1$ offers low throughput. However, its throughput approaches to 100% when iIM-$CM = 8$. This presents the challenge of performing a large number of iterations, therefore, we would need to consider the adoption of a pipelining technique [10]. These results also show that there is a finite number of iIM-CM iterations for this switch to provide high throughput under this traffic pattern.

[2]This is an extended version of CRRD, presented in [5]. A solution for implementation of a scheme that could need time to perform several IM-CM iterations is given in [10].

VI. Conclusions

This paper proposed the MWMD scheme for buffered Clos-network switches, which is based on a maximum weight matching algorithm for input-buffered switches. This paper showed that the combination of MWMD and request queues in a buffered Clos-network switch provides 100% throughput under independent admissible traffic, without allocating any buffers in the second stage and without expanding the internal bandwidth. In addition, we introduced a maximal-weight matching dispatching scheme, MOMD, based on the OCF scheme for single stage switches, which can provide high throughput under a nonuniform traffic pattern, called unbalanced.

References

[1] C. Clos, "A Study of Non-Blocking Switching Networks," Bell Sys. Tech. Jour., pp. 406-424, March 1953.

[2] E. Oki, Z. Jing, R. Rojas-Cessa, and H. J. Chao, "Concurrent Round-Robin Dispatching Scheme in a Clos-Network Switch," IEEE ICC 2001, pp. 107-111, June 2001.

[3] T. Chaney, J. A. Fingerhut, M. Flucke, and J. S. Turner, "Design of a Gigabit ATM switch," Proc. IEEE INFOCOM'97, pp. 2-11, Apr. 1997.

[4] F. M. Chiussi, J. G. Kneuer, and V. P. Kumar, "Low-Cost Scalable Switching Solutions for Broadband Networking: The ATLANTA Architecture and Chipset," IEEE Commun. Mag., pp. 44-53, Dec. 1997.

[5] E. Oki, Z. Jing, R. Rojas-Cessa, and H. J. Chao, "Concurrent Round-Robin-Based Dispatching Schemes for Clos-Network Switches," IEEE/ACM Trans. Networking, vol. 10, no. 6, pp. 830-844, Dec. 2002.

[6] N. McKeown, A. Mekkittikul, V. Anantharam, J. Walrand, "Achieving 100% Throughput in an Input-queued Switch," IEEE Trans. Commun., vol. 47, no. 8, pp. 1260-1267, Aug. 1999.

[7] C. Y. Lee and A. Y. Qruc, "A Fast Parallel Algorithm for Routing Unicast Assignments in Benes Networks," IEEE Trans. Parallel Distributed Sys., vol. 6, no. 3, pp. 329-333, Mar. 1995.

[8] T. T. Lee and S-Y Liew, "Parallel Routing Algorithm in Benes-Clos Networks," Proc. IEEE INFOCOM'96, pp. 279-286, 1996.

[9] K. Pun, M. Hamdi, "Distro: A Distributed Static Round-Robin Scheduling Algorithm for Bufferless Clos-Network Switches," IEEE Globecom 2002, 2002.

[10] E. Oki, R. Rojas-Cessa, H. Jonathan Chao, "PCRRD: A Pipeline-Based Concurrent Round-Robin Dispatching Scheme for Clos-Network Switches," IEEE ICC 2002, vol.4, pp:2121 -2125, 2002.

[11] P. Konghong, M. Hamdi, "Static Round-robin Dispatching Schemes for Clos-network Switches," IEEE HPSR 2002, pp. 329-333, May 2002.

[12] T. Anderson, S. Owicki, J. Saxe, and C. Thacker, "High speed switch scheduling for local area networks," ACM Trans. Comput. Syst., vol. 11, no. 4, pp. 319-352, Nov. 1993.

[13] N. McKeown, Ph. D. Thesis, University of California at Berkeley, May 1995.

[14] S. Iyer, N. McKeown, "Analysis of the Parallel Packet Switch Architecture," IEEE/ACM Trans. Networking, vol. 11, No. 2, pp. 314-324, Apr. 2003.

[15] J. Turner and N. Yamanaka, “Architectural Choices in Large Scale ATM Switches,” IEICE Trans. Commun. vol., E81-B, no. 2, pp. 120-137, Feb. 1998.

[16] R. Rojas-Cessa, E. Oki, Z. Jing, and H. J. Chao, “CIXB-1: Combined Input-One-Cell-Crosspoint Buffered Switch,” *Proceedings of IEEE HPSR 2001*, pp. 324-329, May 2001.